Кнопка (Trema-модуль)

Общие сведения:

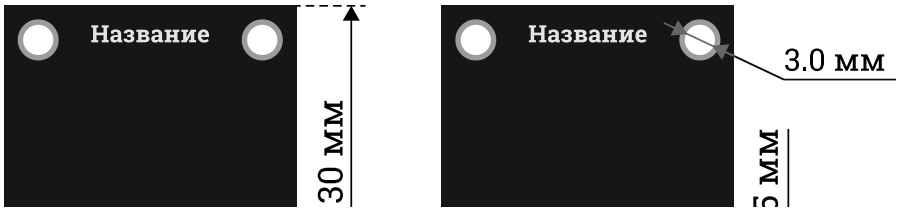
Trema-модуль Кнопка - это тактовая кнопка, которая может служить источником сигналов (команд) для Ваших проектов. Кнопки используются для управления устройствами, подачи команд, осуществления настроек, ввода данных и т.д.

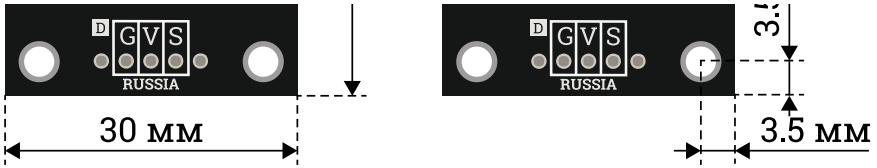
Trema-модуль Кнопка со светодиодом - это аналог *Trema-модуля Кнопка*, но с возможностью управления подсветкой самой кнопки.

Спецификация:

- Рабочее напряжение: до 5,5 В
- Коммутируемый ток: до 50 мА
- Время дребезга при нажатии: < 3 мкс
- Время дребезга при отпускании: < 5 мс
- Сопротивление прижимающего резистора: 10 кОм
- Сопротивление S - V: > 100 МОм (в разомкнутом состоянии)
- Сопротивление S - V: < 100 мОм (в замкнутом состоянии)
- Рабочая температура: -20 ... 70 °C
- Габариты: 30x30x15 (без учёта выводов)

Все модули линейки "Trema" выполнены в одном формате





Подключение:

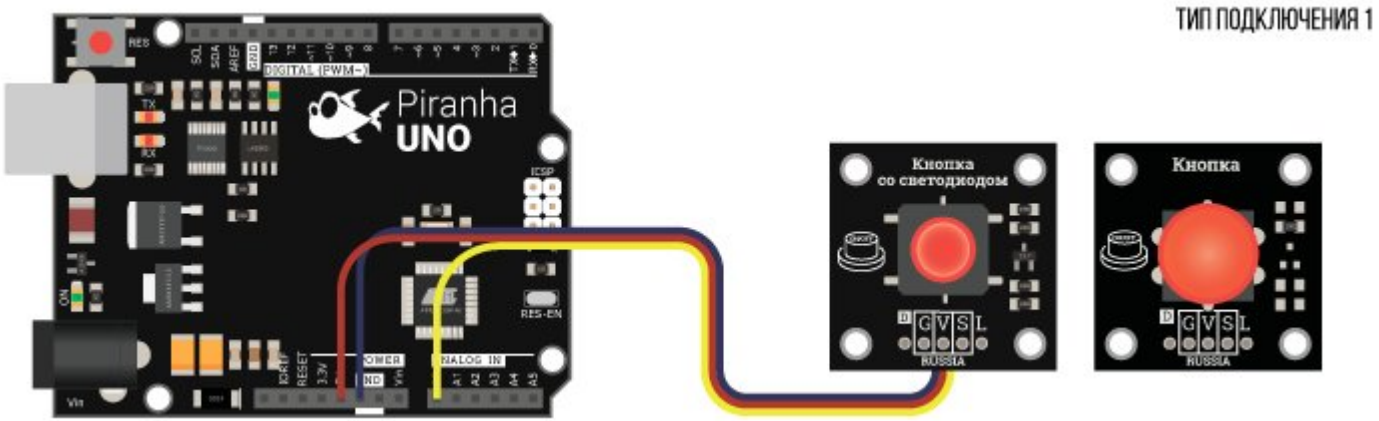
Trema-кнопка и *Кнопка со светодиодом* входят в линейку Trema-модулей, что позволяет подключать их к Arduino через Trema Shield по 3-х и 4-х проводному шлейфу (который идёт в комплекте с кнопкой) без пайки, без дополнительных проводов и переходников.

Trema-кнопку можно подключать к любому выводу Arduino, как цифровому, так и аналоговому.

Модуль удобно подключать 3 способами, в зависимости от ситуации:

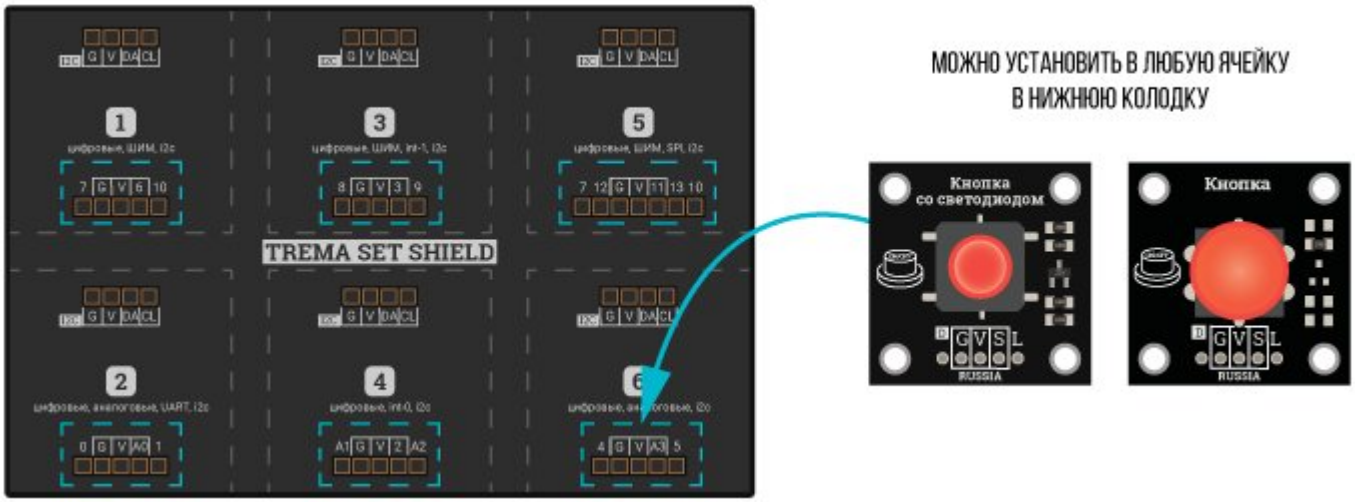
Способ - 1 : Используя проводной шлейф и Piranha UNO

Используя провода «Папа — Мама», подключаем напрямую к контроллеру Piranha UNO.



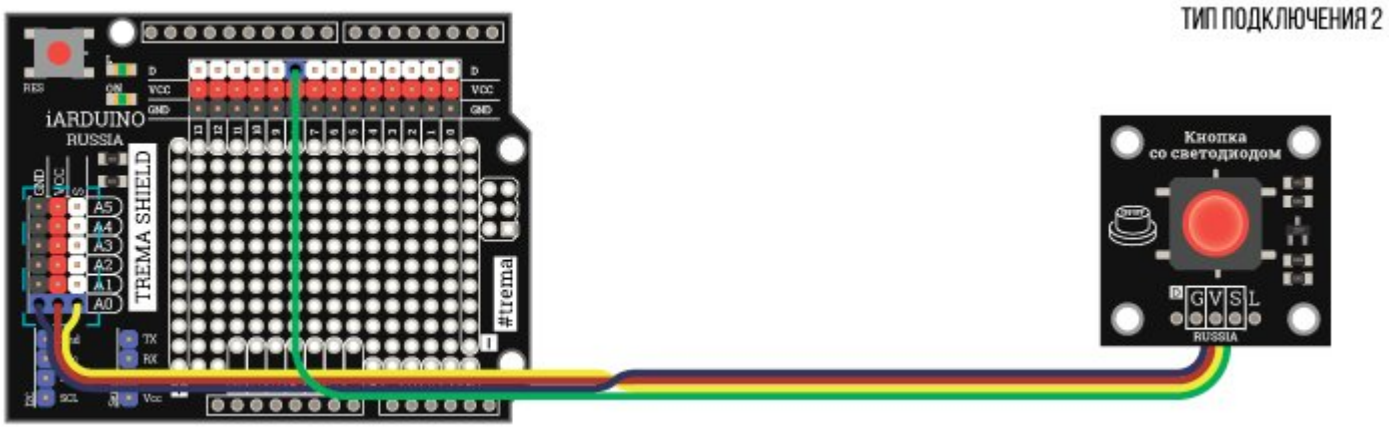
Способ - 2 : Используя Trema Set Shield

Модуль можно подключить к любому из цифровых или аналоговых входов Trema Set Shield.



Способ - 3 : Используя проводной шлейф и Shield

Используя два 2-х или один 3-х проводной шлейф, к Trema Shield, Trema-Power Shield, Motor Shield, Trema Shield NANO и тд.

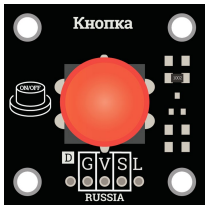


Питание:

Рабочее напряжение питания до 5,5 В постоянного тока

Подробнее о модулях:

Тактовые кнопки предназначены для коммутации электрических цепей и широко используются в радиоэлектронной аппаратуре.



Trema-кнопка имеет **три вывода**: Signal (S), Vcc (V), GND (G). В не нажатом состоянии на выходе S присутствует уровень логического «0» (выход прижат к GND через резистор). В нажатом состоянии на выходе S устанавливается уровень логической «1» (выход соединяется с Vcc).

Для работы с модулем нужно сконфигурировать вывод Arduino, подключённый к выходу модуля, как вход. И считывать состояние логического уровня с данного вывода.

Trema-кнопка со светодиодом имеет **четыре вывода**: Signal (S), Vcc (V), GND (G), Light (L). В не нажатом состоянии на выходе S присутствует уровень логического «0» (выход прижат к GND через резистор). В нажатом состоянии на выходе S

устанавливается уровень логической «1» (выход соединяется с Vcc). Чтобы загорелся светодиод на вывод L необходимо подать уровень логической «1».

При считывании показаний с модулей нужно учитывать такое явление как дребезг контактов. При нажатии или отпускании кнопки, её контакты сначала многократно и неконтролируемо замыкаются и размыкаются по причине упругости их металла, а постоянный логический уровень устанавливается только после окончания дребезга. Это значит, что если 1 раз нажать на кнопку и отпустить её, то алгоритм программы может зафиксировать многократное нажатие на кнопку, если в нём не учитывается подавление дребезга.

Для подавления влияния дребезга на алгоритм скетча, нужно после фиксации изменения логического уровня на выходе кнопки выдержать паузу, равную или превышающую время дребезга.

Примеры:

При работе с кнопкой можно фиксировать её состояния (**нажата** / **отпущена**) и события (**нажимается** / **отпускается**).

Фиксация всех состояний и событий одной кнопки:

```
// Фиксируем НАЖАТИЕ, УДЕРЖАНИЕ, ОТПУСКАНИЕ и НЕ НАЖАТИЕ кнопки:
const uint8_t pinBTN = 2;                           // Определяем номер вывода к которому подключёна кнопка
                                                    //
void setup(){                                       //
    pinMode(pinBTN, INPUT );                        // Конфигурируем вывод к которому подключена кнопка как вход
}                                                   //
                                                    //
void loop(){                                        //
    if(digitalRead(pinBTN)){                        // Если кнопка нажата, то ...
        delay(1);                                   // Выполняем задержку на 1 мс для подавления дребезга при нажатии на кноп
            // КОД ВЫПОЛНЯЕТСЯ ПРИ НАЖАТИИ          // Код в данной части скетча выполняется однократно при нажатии на кнопку
            /* ... КОД ... */                       //
        while(digitalRead(pinBTN)){                 // Если кнопка удерживается, то ...
            // КОД ВЫПОЛНЯЕТСЯ ПРИ УДЕРЖАНИИ        // Код в данной части скетча выполняется постоянно пока нажата кнопка
            /* ... КОД ... */                       //
```

```
        }                                           // Если кнопка отпущена, то ...
        delay(10);                                  // Выполняем задержку на 10 мс для подавления дребезга при отпускании кно
            // КОД ВЫПОЛНЯЕТСЯ ПРИ ОТПУСКАНИИ       // Код в данной части скетча выполняется однократно при отпускании кнопки
            /* ... КОД ... */                       //
    }                                               //
            // КОД ВЫПОЛНЯЕТСЯ ЕСЛИ КНОПКА НЕ НАЖАТА // Код в данной части скетча выполняется постоянно пока кнопка не нажата
            /* ... КОД ... */                       //
}
```

В данном примере каждый участок кода выполняется в зависимости от состояния или события кнопки.

В следующем примере код скетча не разбит на участки, а состояние кнопки хранится в переменной.

Фиксация всех состояний и событий одной кнопки через переменную:

```
const uint8_t  pinBTN = 2;                                          // Определяем номер вывода к которому подключёна кнопка
      bool     valBTN = 0;                                          // Определяем переменную для хранения логического уровня
      uint8_t  evnBTN = 0;                                          // Определяем переменную для хранения состояний и событий кн
                                                                    //
void setup(){                                                       //
    pinMode(pinBTN, INPUT );                                        // Конфигурируем вывод к которому подключена кнопка как вход
}                                                                   //
                                                                    //
void loop(){                                                        //
    valBTN=digitalRead(pinBTN);                                     // Сохраняем логический уровень со входа pinBTN в переменную
    switch(evnBTN){                                                 // Определяем состояния и события кнопки
        case 0:  if( valBTN){evnBTN=1; delay(1);     } break;       // Фиксируем событие:   нажатие    (подавляем дребезг)
        case 2:  if(!valBTN){evnBTN=3; delay(10);    } break;       // Фиксируем событие:   отпускание (подавляем дребезг)
        default: if( valBTN){evnBTN=2;}else{evnBTN=0;} break;       // Фиксируем состояние: удерживается или отпущена
    }                                                               //
                                                                    //
    /* ... КОД СКЕТЧА ... */                                       //
```

```
                                                                    //
    if(evnBTN==0){ /* КНОПКА ОТПУЩЕНА     */ ;}                    // Код выполняется постоянно  при отпущенной кнопке
    if(evnBTN==1){ /* КНОПКА НАЖИМАЕТСЯ   */ ;}                    // Код выполняется однократно при нажимании на кнопку
    if(evnBTN==2){ /* КНОПКА УДЕРЖИВАЕТСЯ */ ;}                    // Код выполняется постоянно  при нажатой кнопке
    if(evnBTN==3){ /* КНОПКА ОТПУСКАЕТСЯ  */ ;}                    // Код выполняется однократно при отпускании кнопки
}
```

Следующий пример является копией данного скетча, только проверяется состояние не одной, а двух кнопок.

Фиксация всех состояний и событий двух (и более) кнопок через переменные:

```
const uint8_t  pinBTN1 = 2, pinBTN2 = 3;                            // Определяем номера выводов к которым подключены кнопки
      bool     valBTN1 = 0, valBTN2 = 0;                            // Определяем переменные для хранения логических уровней
      uint8_t  evnBTN1 = 0, evnBTN2 = 0;                            // Определяем переменные для хранения состояний и событий кн
                                                                    //
void setup(){                                                       //
    pinMode(pinBTN1, INPUT );                                       // Конфигурируем вывод к которому подключена кнопка 1 как вх
    pinMode(pinBTN2, INPUT );                                       // Конфигурируем вывод к которому подключена кнопка 2 как вх
}                                                                   //
                                                                    //
void loop(){                                                        //
    valBTN1=digitalRead(pinBTN1);                                   // Сохраняем логический уровень со входа pinBTN1 в переменну
    valBTN2=digitalRead(pinBTN2);                                   // Сохраняем логический уровень со входа pinBTN2 в переменну
    switch(evnBTN1){                                                // Определяем состояния и события кнопки 1
        case 0:  if( valBTN1){evnBTN1=1; delay(1);      } break;    // Фиксируем событие:   нажатие    (подавляем дребезг)
        case 2:  if(!valBTN1){evnBTN1=3; delay(10);     } break;    // Фиксируем событие:   отпускание (подавляем дребезг)
        default: if( valBTN1){evnBTN1=2;}else{evnBTN1=0;} break;    // Фиксируем состояние: удерживается или отпущена
    }                                                               //
    switch(evnBTN2){                                                // Определяем состояния и события кнопки 2
        case 0:  if( valBTN2){evnBTN2=1; delay(1);      } break;    // Фиксируем событие:   нажатие    (подавляем дребезг)
        case 2:  if(!valBTN2){evnBTN2=3; delay(10);     } break;    // Фиксируем событие:   отпускание (подавляем дребезг)
        default: if( valBTN2){evnBTN2=2;}else{evnBTN2=0;} break;    // Фиксируем состояние: удерживается или отпущена
```

```
    }                                                                   //
                                                                        //
    /* ... КОД СКЕТЧА ... */                                           //
                                                                        //
    if(evnBTN1==0){ /* КНОПКА 1 ОТПУЩЕНА     */ ;}                    // Код выполняется постоянно  при отпущенной кнопке
    if(evnBTN1==1){ /* КНОПКА 1 НАЖИМАЕТСЯ   */ ;}                    // Код выполняется однократно при нажимании на кнопку
    if(evnBTN1==2){ /* КНОПКА 1 УДЕРЖИВАЕТСЯ */ ;}                    // Код выполняется постоянно  при нажатой кнопке
    if(evnBTN1==3){ /* КНОПКА 1 ОТПУСКАЕТСЯ  */ ;}                    // Код выполняется однократно при отпускании кнопки
                                                                        //
    if(evnBTN2==0){ /* КНОПКА 2 ОТПУЩЕНА     */ ;}                    // Код выполняется постоянно  при отпущенной кнопке
    if(evnBTN2==1){ /* КНОПКА 2 НАЖИМАЕТСЯ   */ ;}                    // Код выполняется однократно при нажимании на кнопку
    if(evnBTN2==2){ /* КНОПКА 2 УДЕРЖИВАЕТСЯ */ ;}                    // Код выполняется постоянно  при нажатой кнопке
    if(evnBTN2==3){ /* КНОПКА 2 ОТПУСКАЕТСЯ  */ ;}                    // Код выполняется однократно при отпускании кнопки
}
```

В любом из указанных скетчей можно использовать не все участки кода, а только те, которые требуются Вам. Ниже описаны примеры, реагирующие только на одно действие.

Управляем подсветкой кнопки при каждом нажатии на неё:

Подключите вывод кнопки к пину 5, а вывод подсветки к пину 6.

```
const uint8_t pinBTN = 5;                           // Определяем номер вывода к которому подключёна кнопка
const uint8_t pinLED = 6;                           // Определяем номер вывода светодиода, который установлен на кнопке
void setup(){
    pinMode(pinBTN, INPUT );                        // Конфигурируем вывод кнопки как вход
    pinMode(pinLED, OUTPUT);                        // Конфигурируем вывод светодиода как выход
}
void loop(){
    if(digitalRead(pinBTN)){delay(1);               // Если кнопка нажимается, то подавляем дребезг
        digitalWrite(pinLED, !digitalRead(pinLED)); // и меняем состояние на выходе светодиода
        while(digitalRead(pinBTN)){}                // Если кнопка удерживается, то ничего не делаем
```

```
        delay(10);                                  // Если кнопка отпускается, то подавляем дребезг
    }
}
```

Светодиод будет менять своё состояние (включён/выключен) при каждом нажатии на кнопку.

Управляем подсветкой кнопки при каждом отпускании её:

```
const uint8_t pinBTN = 5;                           // Определяем номер вывода к которому подключёна кнопка
const uint8_t pinLED = 6;                           // Определяем номер вывода светодиода, который установлен на кнопке
void setup(){
    pinMode(pinBTN, INPUT );                        // Конфигурируем вывод кнопки как вход
    pinMode(pinLED, OUTPUT);                        // Конфигурируем вывод светодиода как выход
}
void loop(){
    if(digitalRead(pinBTN)){delay(1);               // Если кнопка нажимается, то подавляем дребезг
        while(digitalRead(pinBTN)){}                // Если кнопка удерживается, то ничего не делаем
        digitalWrite(pinLED, !digitalRead(pinLED)); // Если кнопка отпускается, то меняем состояние на выходе светодиода
        delay(10);                                  // и подавляем дребезг при отпускании кнопки
    }
}
```

Светодиод будет менять своё состояние (включён/выключен) при каждом отпускании кнопки.

Включаем подсветку кнопки при её удержании дольше 2 секунд:

```
const uint8_t pinBTN = 5;                           // Определяем номер вывода к которому подключёна кнопка
const uint8_t pinLED = 6;                           // Определяем номер вывода светодиода, который установлен на кнопке
uint8_t       timeLED;                              // Объявляем переменную для хранения времени удержания кнопки
void setup(){
    pinMode(pinBTN, INPUT );                        // Конфигурируем вывод кнопки как вход
    pinMode(pinLED, OUTPUT);                        // Конфигурируем вывод светодиода как выход
```

```
}
void loop(){
    if(digitalRead(pinBTN)){ timeLED=0;             // Если кнопка нажимается, то сбрасываем счётчик времени
        while(digitalRead(pinBTN)){                 // Если кнопка удерживается, то ...
            if(timeLED>=200){                       // Если значение счётчика >= 200
                digitalWrite(pinLED, HIGH);         // Включаем светодиод
            }else{timeLED++;}                       // Иначе, приращаем счетчик времени
            delay(10);                              // Устанавливаем задержку на 10 мс
        }
        digitalWrite(pinLED, LOW);                  // Если кнопка отпускается, то выключаем светодиод
    }
}
```

Светодиод будет включаться, только если кнопка удерживается дольше 2 секунд. А выключаться будет сразу при отпускании кнопки.

При удержании кнопки, постоянно выполняется цикл **while** с задержкой в теле цикла на **10 мс**. С каждым проходом данного цикла, счетчик **timeLED** увеличивается на единицу. После того как значение счетчика **timeLED** станет **>= 200** (200*10 мс = 2 сек.) включается светодиод, а счетчик перестанет увеличиваться. Как только кнопка будет отпущена, произойдет выход из цикла **while** и светодиод погаснет.

В отличии от предыдущих скетчей, данный скетч использует только одну функцию **delay()** которая выполняется внутри цикла **while** постоянно, пока нажата кнопка, а следовательно, подавляет дребезг и при нажатии и при отпускании кнопки.

Применение:

- Управление устройствами;
- Пульты управления;
- Подача управляющих команд системе и пр.